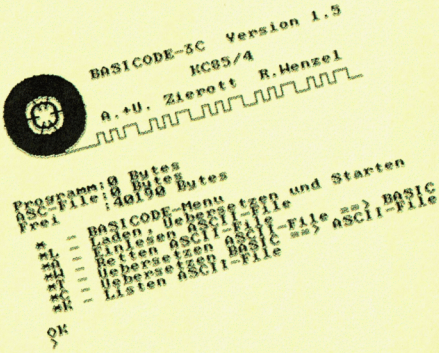
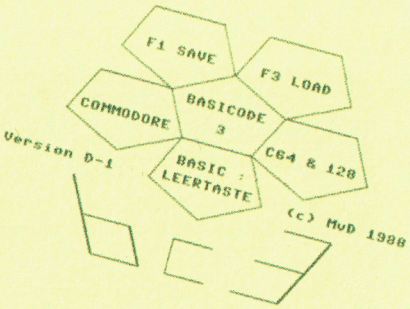
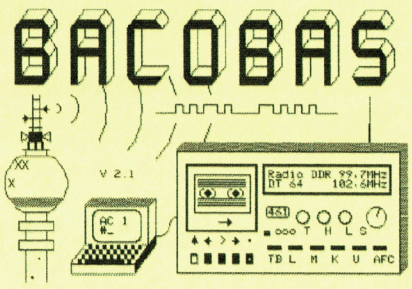
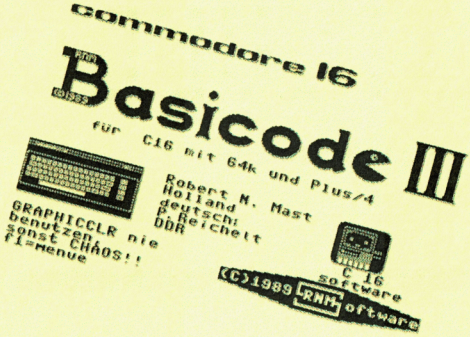

BASICODE-3C Version 1.5
KC85/4
A.+U. Zierott R.Wenzel
Programm:0 Bytes
ASC-File:0 Bytes
Frei :40190 Bytes
* - BASICODE-Menu
*L - Laden, Uebersetzen und Starten
*A - Einlesen ASCII-File
*W - Retten ASCII-File
*T - Uebersetzen ASCII-File ==> BASIC
*C - Uebersetzen BASIC ==> ASCII-File
*K - Listen ASCII-File
OK

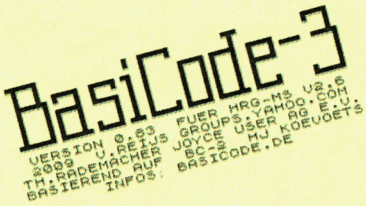

F1 SAVE
F3 LOAD
COMMODORE
BASICODE 3
C64 & 128
BASIC : LEERTASTE
Version D-1
(c) MvD 1988

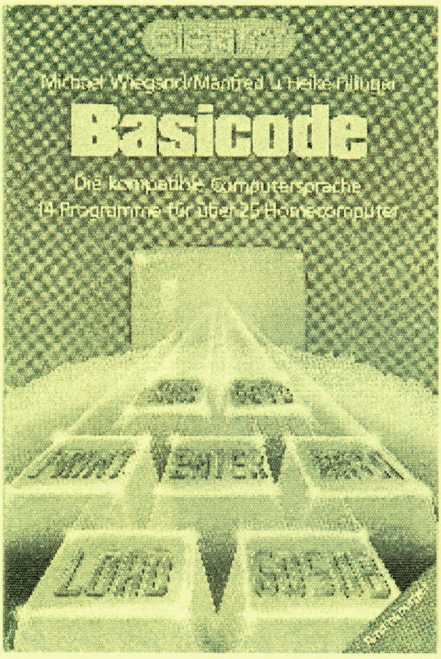
Dragon → VC 20 → Apple ↓↓ → Colour Genie
BASICODE
Version 3-C
ein Esperanto
für Computer
PET 2001 → ZX Spectrum
CPC 464 → TRS 80 1 → KC 85/3 → ZX 81
KC↔ C plus ↓ → ...

BasiCode-3C
für CPC 464, 664, 6128 und KC compact
welches BasiCode-Programm laden ;
zum Laden im BasiCode-eigenen Format
BasiCode-3 Subs V4.1 19-2 '89
(C) S.Simons, M.van der Meer
auf Color-Betrieb (BasiCode-3C)
angepaßt 28-8 '04 Th.Rademacher


BACOBAS
V 2.1
AC 1
Radio DDR 99.7MHz
DT 64 102.6MHz
T H L S
TB L M K U AFC


commodore 16
Basicode III
für C16 mit 64k und Plus/4
Robert M. Mast
Holland
deutsch:
P.Reichelt
DDR
GRAPHICCLR nie benutzen, sonst CHAOS!!
f1=menue
C 16 software
(C)1989 RNM software

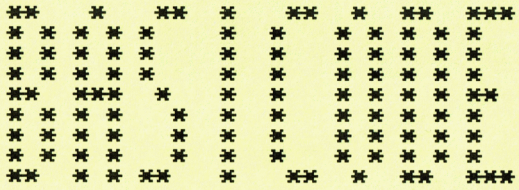

BasiCode-3
VERSION 0.83 FUER HRG-MS V2.6
2009 V.REIJS GROUPS.YAHOO.COM
TH.RADEMACHER JOYCE USER AG E.V.
BASIEREND AUF BC-2 MJ KOEVOETS
INFOS: BASICODE.DE



1984 ...

(c) 2016, 2017 Rob Hagemans

... 2017 ...



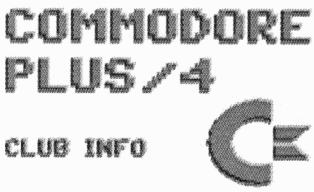
---==[2017]==---

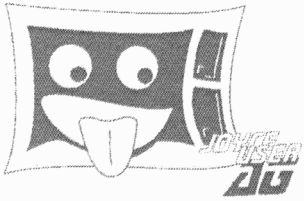
-- drag and drop to load --

... still alive!

Inhalt

1.	Einleitung	2
2.	Zeilenverwendung	4
3.	Variablen und Felder	5
4.	Befehle	7
5.	Die Subroutinen des Bascoders	11
6.	Textmodus	12
7.	Grafikmodus	16
8.	Farben	17
9.	Sound	19
10.	Arbeit mit Dateien	20
11.	Drucken (außer Grafik)	22
12.	Weitere Bascoder-Routinen	24
13.	Das BasiCode-Protokoll	27
14.	weblinks	28
15.	Ein Mini-Wörterbuch	29
16.	Literatur	29





```
1981 BasiCode-1  einheitliches Kassettenaufzeichnungsverfahren für
     unterschiedliche Computer, Verwendung von ASCII-Listings statt
     Token, geringe Störanfälligkeit, Übertragung sogar auf
     Mittelwelle möglich, eingeschränkter Befehlsvorrat
1984 BasiCode-2  weitere Befehle nutzbar mittels Ersatz durch
     (computerspezifische) GOSUB-Routinen
1987 BasiCode-3  einkanaliger Sound und monochrome Pixelgrafik sowie
     Arbeit mit Dateien kommen hinzu
1991 BasiCode-3C  Verwendung von acht Farben für Text und Grafik,
     Textdump, Abfrage von (sofern vorhandenen) Funktionstasten
```

BasiCode-Versionen

1. Einleitung

BasiCode ist eine Basic-Variante, die sich gut durch den Begriff "Computer-Esperanto" charakterisieren läßt.
Ab 1980 kamen immer mehr Computer auf den Markt, die für den privaten Anwender erschwinglich waren. Gewöhnlich waren sie mit der anfängerfreundlichen Programmiersprache Basic ausgestattet, doch in mindestens so viel gegenseitig inkompatiblen Dialekten wie es Hersteller gab.
Aus dem Wunsch von Hobbyisten nach Erfahrungsaustausch über Systemgrenzen hinweg wurde in den Niederlanden mit BasiCode-1 zunächst ein vereinheitlichtes Kassettenaufzeichnungsverfahren entwickelt.
Man schränkte den Befehlsumfang auf die Befehle ein, die von allen Heimcomputern verstanden wurden. Weil diese Beschränkung kein befriedigender Zustand war, kam man auf den Gedanken, weitere Befehle, die bei gleicher Wirkung nur unterschiedliche Namen bei den verschiedenen Computern hatten, als GOSUB-Routinen zu realisieren.
Für den einzelnen Computertyp wurde nun ein individuelles Grundprogramm, der sogenannte Bascoder, erstellt, der in diesen GOSUB-Routinen die computereigenen Befehle ausführte.
Solche Bascoder entstanden für nahezu jeden Heimcomputer und darüber hinaus auch für Basic auf CP/M- und auch DOS-Computern.

Größere Popularität erreichte BasiCode außer in seinem Herkunftsland, den Niederlanden, in der DDR, weil dort ein großer Mangel auch an Software herrschte, und erlebte seine Blütezeit Ende der 80er Jahre, besonders faszinierend war zu jener Zeit die Verbreitung von BasiCode-Programmen über den Rundfunk, sogar über mehrere 100 km hinweg auf Mittelwelle.
Mit der immer stärkeren Verdrängung der Heimcomputer durch "große PCs" in den Privathaushalten geriet auch BasiCode weitestgehend in Vergessenheit. Zwar haben sich seine Datenaustauschmöglichkeiten erweitert, man ist keinesfalls mehr nur auf Kassetten beschränkt, sondern es stehen außer Disketten, SD-Cards und USB-Sticks auch DFÜ und Internet zur Verfügung, aber die vergleichbare äußerliche Schlichtheit der Programme lockt heute keinen 3D-Grafik-, Stereo-Sound-, MHz- und Gigabyte-verwöhnten Konsumenten von Spiele-CDROMs mehr hinter dem Ofen hervor, sodaß BasiCode heutzutage nur eine interessante historische Entwicklung ist, die höchstens noch von solchen Nostalgikern wahrgenommen wird, die ohnehin nicht von ihren Geräten der "Computer-Steinzeit" lassen können.

Rechtliches:

Um die Aktivitäten zur Weiterentwicklung von BasiCode zu koordinieren, hatten sich die Autoren dieses Programmsystems in einer Stiftung zusammengeschlossen, der NOS (Nederlandse omroep Stichting) und zur Förderung des BasiCode-Gedankens auf persönliche materielle Vergütung verzichtet. Programme anderer Autoren darf man zwar weiterentwickeln, aber die neue Version nicht ohne Genehmigung des Autors verbreiten. Der Name des Autors darf bei einer Kopie nicht entfernt werden. Auch Übersetzungen bedürfen der Genehmigung des Au-

tors. Programme in BasiCode dürfen nicht verkauft oder in sonstiger Weise zur persönlichen Bereicherung genutzt werden. Wenn sich die NOS auch 1991 aufgelöst hat, sollte es trotzdem eine Frage des Anstands sein, diese Maßgaben weiter zu respektieren.

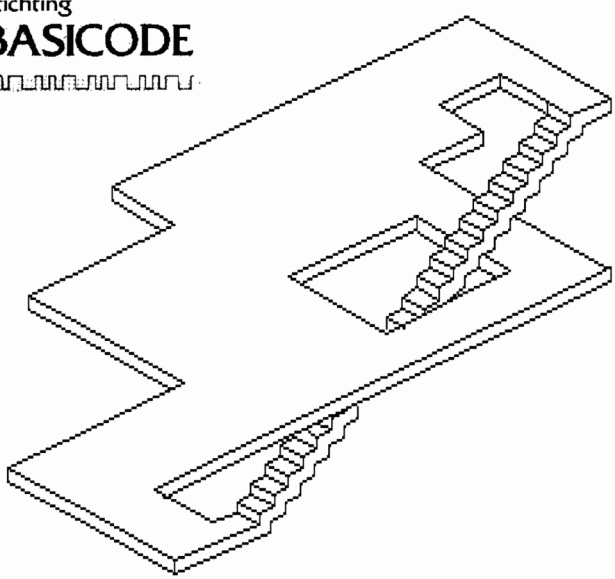
In der vorliegenden Broschüre sollen die Festlegungen erläutert werden, die unumgänglich sind, damit BasiCode-Programme auch wirklich auf allen beteiligten Computern genutzt werden können, wie unterschiedlich auch die Möglichkeiten sind, die die einzelnen Computer zunächst bieten.

Kursiv dargestellte Passagen kennzeichnen neue Eigenschaften der farbtauglichen jüngsten Version. Für monochrome Computer wie Sinclair ZX81 oder Amstrad PCW (Schneider JOYCE) wurden die Neuerungen der Version 3C nach Möglichkeit umgesetzt.

BASICODE 2

nederlandse omroep stichting

stichting BASICODE



2. Zeilenverwendung

Die Zeilen von 0 bis 999 sind dem computerspezifischen Bascoder vorbehalten. Auf manchen Computern sind sie in Maschinencode realisiert und daher nicht wie normale Basic-Zeilen listbar. Die hier liegenden GOSUB-Routinen werden in Kapitel 5 zusammengefaßt und in den nachfolgenden Kapiteln im Detail beschrieben.
Das (auf andere Computer übertragbare) BasiCode-Programm nutzt die Zeilen 1000 bis 32767. Zeilennummern unter 1000 sind nicht zugelassen! Die Länge einer Zeile darf einschließlich Zeilennummer, Leerzeichen und Zeichen für Zeilenende höchstens 60 betragen. Eine Programmzeile kann mehrere Anweisungen - getrennt durch ":" (Doppelpunkt) - enthalten.
Es muß stets eine Zeile 1000 und eine Zeile 1010 vorhanden sein. Zeile 1000 hat folgende vorgeschriebene Form: **1000 A=wert:GOTO 20:REM programmname**
Mit der Variablen A wird (für die Computer, bei denen das erforderlich ist) festgelegt, wie groß der Speicherbereich sein soll, der für Stringvariablen reserviert wird. Anschließend wird das Programm in Zeile 20 des Bascoders fortgesetzt, wo die Initialisierung des Computers für die BasiCode-Betriebsart durchgeführt wird:

- Löschen des Bildschirms,
- Löschen der Variablen,
- Wahl des Text-Modus,
- Setzen der Bildschirmfarben

Die Code-Ziffern werden den Variablen CC(0) und CC(1) zugewiesen; als Default-Einstellung gilt: CC(0)=7 -> Zeichenfarbe – Weiß, CC(1)=0 -> Hintergrund - Schwarz Bei Start des Programms werden diese Werte mit der Subroutine 100 übernommen und die Farben entsprechend gesetzt

- Deklarieren und Initialisieren der Variablen HO, VE, HG, VG und SV.

Von dort erfolgt ein Rücksprung nach Zeile 1010.

Für das Beenden des BasiCode-Programms ist der Befehl GOTO 950 vorgeschrieben.
Dort wird der Computer definiert in seine reguläre Basic-Betriebsart zurückversetzt, der Bildschirm wird gelöscht, BasiCode-Programm und -Subroutinen bleiben erhalten.
Für die Nutzung der Zeilen ab 1000 gibt es weitere Festlegungen, die im nachfolgenden Textkasten aufgeführt sind.

```
 1010 ... 19999   Hauptprogramm ... GOTO 950 (Ende)

20000 ... 24999   Computerspezifische Unterprogramme, die in BasiCode
                  nicht erlaubte Anweisungen enthalten

25000 ... 29999   DATA-Zeilen

30000 ... 31999   REM-Zeilen: Hinweise, Bemerkungen

32000 ... 32767   REM-Zeilen: Autor, Computer, Datum
```

Zeilenverwendung

3. Variablen und Felder in BasiCode

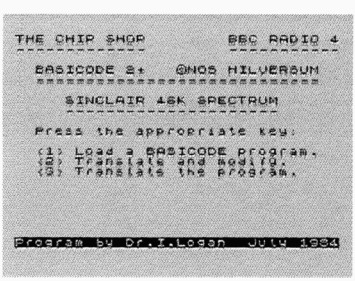
Allen im Programm verwendeten Variablen ist vor ihrem ersten Aufruf ein Wert zuzuweisen - nicht alle beteiligten Computer sind in der Lage, Variablen implizit zu initialisieren. Die Variablen HO, VE, HG, VG und SV werden durch den Sprung nach Zeile 20 initialisiert.
Numerische Variablen sind vom Typ "real" und haben einfache Genauigkeit (sechs gültige Ziffern), Integer-Variablen (z.B. als Schleifenzähler) sind nicht nutzbar.
Eine Stringvariable wird durch den Zusatz "$" nach dem Namen gekennzeichnet; sie kann 255 Zeichen lang sein.
Logische Werte werden je nach Computer unterschiedlich repräsentiert (z.B. "wahr" => "+1" oder "-1"). Eine logische Variable darf deshalb nicht Gegenstand arithmetischer Operationen sein. Vergleiche, die "wahr" oder "falsch" ergeben, sind in Klammern zu setzen, um die Abarbeitungsfolge zu gewährleisten.
Felder (arrays) sind vor Gebrauch in einem Programm zu dimensionieren. Ein wiederholtes Ausführen der Dimensionierung ist zu vermeiden. Mit einer DIM-Anweisung können mehrere Felder gleichzeitig dimensioniert werden. Zugelassen sind ein- oder zweidimensionale Felder (Listen oder Tabellen). Mit der Dimensionierung werden gleichzeitig die Elemente auf "0" bzw. "leer" gesetzt. Es kann nicht davon ausgegangen werden, daß der Aufruf einer Feldvariablen im Programm automatisch ein Feld mit 11 Elementen dimensioniert. Auch Felder mit weniger als 11 Elementen sind zu dimensionieren (z.B. DIM A(4)). Die Zählung der Feldelemente beginnt bei "0".
Variablen- und Feldernamen sind maximal zwei Zeichen lang (Stringvariablen einschließlich Kennung "$" also maximal drei):

- Großbuchstabe
- Großbuchstabe, Großbuchstabe oder
- Großbuchstabe, Ziffer, keine Sonderzeichen wie "!", "%", "#" usw.

Um Konflikte mit Schlüsselwörtern und Systemvariablen mancher Computer zu vermeiden, sind eine Reihe von Variablennamen verboten:

```
AS, AT, DI, DI$, DO, DO$, DS, DS$, EI, EI$, EL, ER, FN, GO, GO$, GR, HC,
IF, LN, MA, MP, PI, SQ, SQ$, ST, ST$, TI, TI$, TO, TO$
```

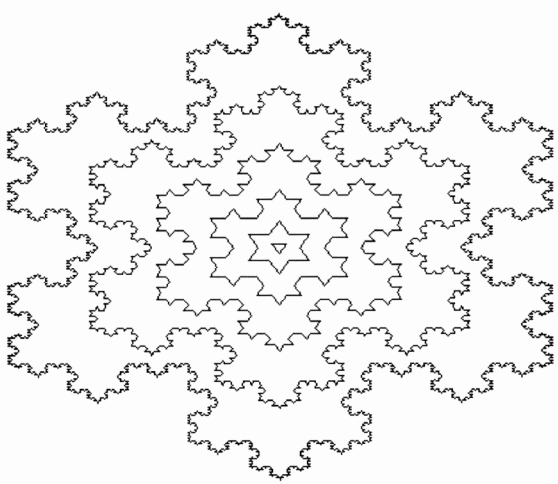
verbotene Variablennamen



Außerdem sind Variablennamen mit dem Anfangsbuchstaben "O" der Verwendung in den GOSUB-Routinen unterhalb der Zeile 1000 vorbehalten.
Bestimmte Variablennamen sind für die Übergabe von Parametern an die Bascoder-Routinen reserviert.

```
HO,VE   - Cursor-Positionierung Textbetrieb (110, 120)
        - Auslesen des Textschirms (220)
        - Cursor-Positionierung Grafik-Betrieb (620 ... 650)
IN, IN$ - Tastaturabfrage (200, 210, 450)
IN      - Auslesen Textschirm (220)
        - Status-Variable externe Speicher (540, 550, 560)
IN$     - Lesen von externen Speichern (550)
SR, SR$ - Bilden einer Stringvariablen (300, 310)
SR$     - String-Ausgabe auf Bildschirm (150, 650)
        - Drucker (350)
        - Externe Speicher (560)
        - Klein- => Groß-Buchstaben (330)
CN      - Formatierung numerischer Ausgaben (310)
        - Zeichenfarbe bei grafischem Betrieb (620, 630, 650)
        - Unterscheidung Groß-/ Kleinbuchstabe (220)
CC      - momentan für CN Festgelegte Farbe bei grafischem
          Betrieb (620, 630, 650)
CT      - Formatierung numerischer Ausgaben (310)
NF, NF$ - Externes Speichern von Daten-Files (500 ... 580)
FR      - Freier Speicherplatz (270)
        - Wirksamkeit der Stop-Taste (280)
SD      - Warte-Routine (450)
        - Musik (400)
SP, SU  - Musik (400)
RU      - Zufallsvariable (260)
HG, VG  - Grafischer Cursor, Bildpunkte (620, 630, 650)
```

Verwendung der reservierten Variablen



=>

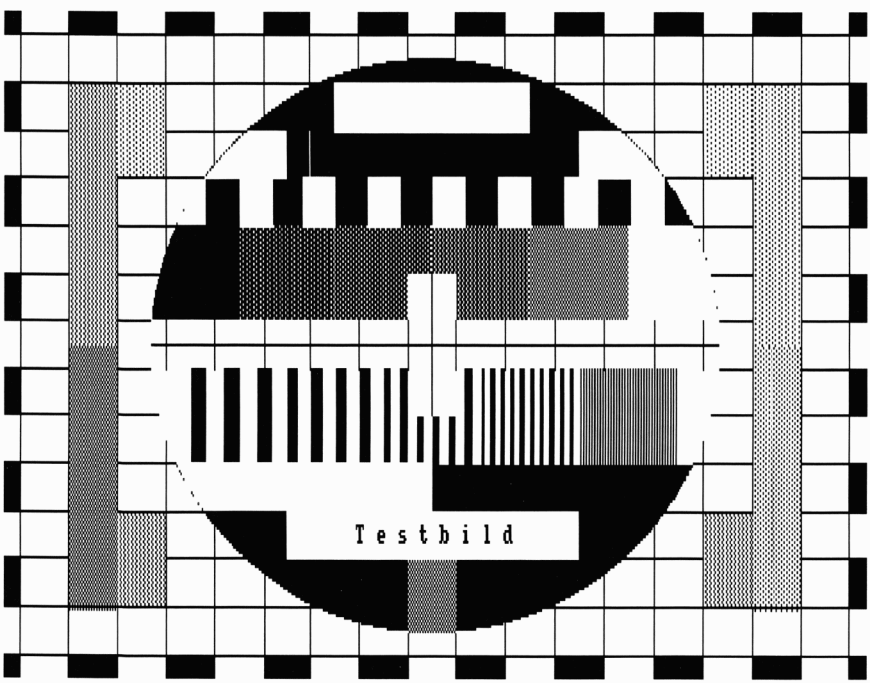
4. Befehle

Folgende Befehle dürfen in BasiCode-Programmen verwendet werden:

```
DATA     DEF FN  DIM    FOR   GOSUB   GOTO    IF      INPUT  LET   NEXT
ON       PRINT   READ   REM   RESTORE RETURN  STEP    TAB    THEN  TO

Funktionen und Operatoren:

ABS      ASC     ATN    CHR$  COS     EXP     INT     LEFT$  LEN   LOG
MID$     RIGHT$  SGN    SIN   SQR     TAN     VAL

AND      OR      NOT

 ^  Exponentation
 *  Multiplikation
 /  Division
 +  - Addition
    - Verkettung (concatenation) von Stringvariablen
 -  Subtraktion

 =    <>    \
 <    >      > Vergleichsoperatoren
 <=   >=    /

Die Folge der Zeichen der Vergleichsoperatoren "<=", ">="
und "<>" ist vorgeschrieben.
                                                  erlaubte Befehle
```

Bei einigen davon sind jedoch zusätzliche Hinweise zu beachten:
Für die Winkelfunktionen ist das Argument im Bogenmaß anzugeben.
Die logischen Operatoren **AND** und **OR** dürfen in BasiCode nicht zur bitweisen Verknüpfung von Zahlen, z.B. A=5 AND 7 verwendet werden.
Bei der Verwendung der Funktionen **ASC** und **CHR$** (der Umkehrfunktion von ASC) ist Vorsicht geboten, da manche Computer den Zeichensatz abweichend vom ASCII codieren (z.B. C64).
Neben den eingebauten Standardfunktionen können auch vom Benutzer definierte Funktionen nach ihrer Definition mit **DEF FN**name verwendet werden. Für den Funktionsnamen gelten die Festlegungen wie für Variablennamen (also max. zwei Zeichen). Die Definition muß in eine Zeile (60 Zeichen) passen. Sie ist nur in der einfachen Form mit einer Variablen und nicht rekursiv anwendbar. Zeichenkettenfunktionen durch den Benutzer zu definieren, ist in Basi-Code-Programmen nicht möglich.
DATA: Bis zum Zeilenende folgen der DATA-Anweisung, durch Komma getrennt, Zahlen oder Zeichenketten. Zeichenketten müssen in Anführungszeichen gesetzt werden. In einer DATA-Zeile dürfen keine weiteren Basic-Anweisungen (auch nicht REM) stehen.
DIM s. S. 5
FOR ... TO ... STEP ... NEXT: Die Schleife wird mindestens einmal durchlaufen. Ohne STEP zahl ist die Schrittweite 1. Die Schleife darf nur an einer einzigen Stelle verlassen werden. Aus der Schleife darf nicht herausgesprungen werden - vorzeitiges Verlassen durch: Laufparameter = Endwert, Sprung zum NEXT.

GOSUB: Zeilennummer darf keine Variable sein, Zeilennummer muß existieren.
GOTO: siehe GOSUB, Ausnahme 20 und 950.
IF ... THEN GOSUB oder **IF ... THEN zeilennummer**. **ELSE** nicht nutzbar
INPUT: nur für eine Variable oder Zeichenkette. Zeichenketten dürfen keine Kommentare und keine Doppelpunkte enthalten. Eine Prompt-Zeichenkette ist nicht zulässig (muß durch vorherige PRINT-Ausgabe erzeugt werden).
LEFT$: Anzahl der zu isolierenden Zeichen darf nicht 0 sein.
LET: kann weggelassen werden.
LOG(argument) gibt den natürlichen Logarithmus zur Basis e (= 2.718...) zurück.
Einzelne BASIC-Dialekte geben über LOG als Wert der Funktion den Logarithmus zur Basis 10. Es wird zwischen LOG und LN unterschieden. Der Bascoder gleicht dies aus.
MID$(A$, N) oder **MID$(A$, M, N)**: Der numerische Ausdruck (M, N) kann einen Wert von 1 ... 255 annehmen; der Wert "0" führt zu einer Fehlermeldung.
NEXT: siehe FOR. Nach NEXT muß die zugehörige Variable stehen, mehr als eine Variable ist nicht zulässig. Für jedes FOR nur ein NEXT zulässig.
NOT: siehe AND
ON: die Variable darf nur die zulässigen Werte annehmen, also von 1 bis zur Anzahl der nach GOTO bzw. GOSUB stehenden Adressen

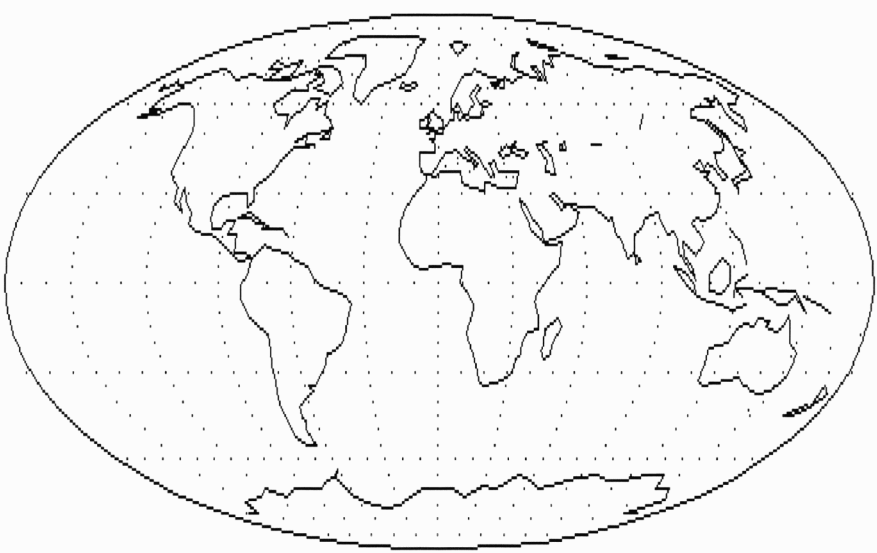


OR: siehe AND
PRINT: Formatierung nur mit ",", ";", und TAB. Empfehlung, "," und TAB durch GOSUB 110 zu ersetzen. Nicht durch "?" ersetzbar.
READ: liest eine oder mehrere Variablen aus den DATA-Zeilen, Variablentyp muß korrekt sein.
REM: nicht durch "'" oder "!" ersetzbar, kein ":" in REM-Zeile.
RESTORE: nur ohne Zeilennummer
**RIGHT$**: s. LEFT$.
RUN: nur ohne Zeilennummer verwendbar.
TAB: TAB(0) nicht erlaubt. Sicherer GOSUB 110.
VAL: Da die VAL(A$)-Funktion in unterschiedlichen Dialekten unterschiedliche Wirkungen hat, ist sicherzustellen, das A$ rein numerisch ist.

Befehle, die es in nicht allen Basic-Dialekten gibt, sind in BasiCode verboten, einige von ihnen sind jedoch durch Subroutinen des Bascoders ersetzbar (s. Kap. 5).

!	CLOSE	INP	NULL	PSET	TROFF
AT	CLS	INPUT#	OPEN	RND	TRON
BEEP	CONT	INSTR	OUT	RANDOMIZE	USR
BLOAD	CSAVE	JOYST	PAPAR	RENUMBER	VGET$
BORDER	CSRLIN	KEY	PAUSE	RUN	VPEEK
BYE	DEEK	KEYLIST	PEEK	SOUND	VPOKE
CALL	DOKE	LINE	PI	SPC	WAIT
CIRCLE	END	LINES	POKE	STOP	WIDTH
COLOR	FRE	LIST#	POS	STRING$	WINDOW
CLEAR	INK	LOAD#	PRESET	STR$	
CLOAD	INKEY	LOCATE	PRINT#	SWITCH	

verbotete Befehle



Weitere Basicbefehle, die spezifisch vorhanden sind und hier nicht aufgeführt wurden, sollten auch nicht genutzt werden,um eine universelle Austauschbarkeit zu gewährleisten.
Die numerischen Funktionen

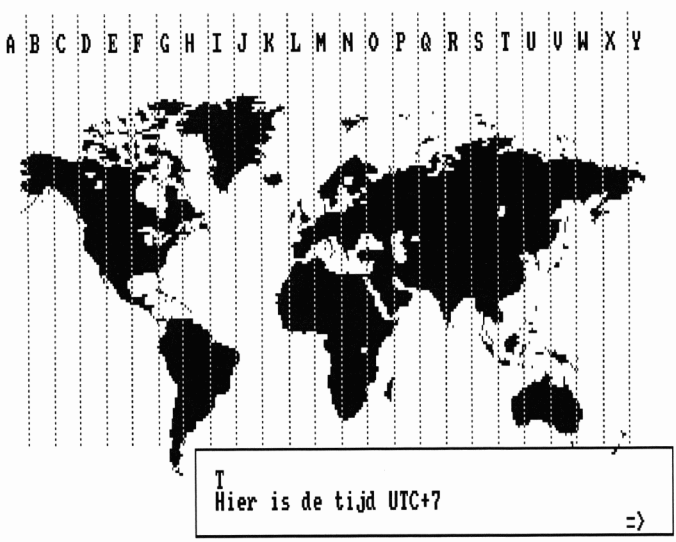
- **MEM** oder
- **FRE**(parameter)

zur Angabe des freien Speicherplatzes dürfen in BasiCode nicht verwendet werden. Dazu dient allein die Subroutine 270 mit gleichzeitiger "garbage collection" und die Abfrage des Wertes der Variablen FR.

Die Bildung von Pseudozufallszahlen geschieht in BasiCode-Programmen allein über die Subroutine 260 und die Variable RV. Funktionsaufrufe oder Kommandos wie:

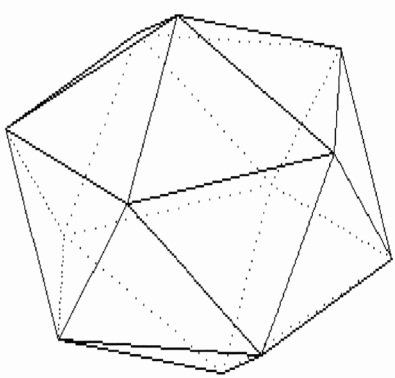
- **RND(parameter)**,
- **RANDOMIZE, RAND**

sind nicht zugelassen.
Die Funktion **STR$(A)** ist in BasiCode verboten; an ihrer Stelle ist die Subroutine 300 mit den Variablen SR bzw. SR$ aufzurufen.



5. Die Subroutinen des Bascoders

GOSUB	Wirkung	Variable	Seite
100	Textbetrieb, Schirm löschen	--	12
110	Positionieren des Cursors	HO, VE	13
120	Registrieren der Cursorposition	HO, VE	13
150	Bildschirm – reverse Darstellung	SR$	13
200	Tastaturabfrage	IN$, IN	14
210	Warten auf Tastendruck	IN$, IN	14
220	Zeichen auf Bildschirmposition	HO, VE, IN	15
250	Akustisches Signal	--	19
260	Pseudozufallszahl	RV	24
270	Freier Speicherplatz	FR	24
280	Abschalten der Stoptaste	FR	24
300	Wandeln in Stringvariable	SR, SR$	25
310	Formatieren numer. Variablen	SR, SR$, CT, CN	25
330	Wandeln von Klein- in Großbuchst.	SR$	26
350	Druckerausgabe	SR$	22
360	Drucker: Wagenrücklauf, Zeilenvorschub	–	22
400	Tonerzeugung	SP, SD, SV	19
450	Warteroutine	SD, IN, IN$	26
500	/ Öffnen des Kanals	NF$, NF	20
540	Peripherer / Lesen	IN$, IN	21
560	Speicher \ Schreiben	SR$, IN	21
580	\ Schließen des Kanals	NF	21
600	Grafischer Betrieb, Schirm löschen	–	16
620	Punkt (Pixel) setzen bzw. Löschen	HO, VE, CN	16
630	Linie zeichnen bzw. Löschen	HO, VE, CN	16
650	Text im Grafikbetrieb ausgeben	SR$, HO, VE	16



6. Textmodus

GOSUB 100

Die Subroutine 100 löscht den Bildschirm und setzt den Computer in den Textbetrieb, sofern vorher über die Subroutine 600 der Grafik-Betrieb eingeschaltet war. *Der Hintergrund erhält die Farbe, die in CC(1) festgelegt ist und die Schrift wird in der Farbe entsprechend dem Inhalt von CC(0) dargestellt. Diese Farben gelten im Ablauf des Programms bis zu einer Änderung von CC(0) und/oder CC(1) und der nächstfolgenden GOSUB 100-Anweisung.* Mit dem Start eines Programms (Initialisierung über den Sprung GOTO 20) wird das Unterprogramm 100 automatisch aufgerufen.

Werte, die Variablen vor Aufruf des Unterprogramms zugewiesen wurden, werden nicht verändert. Dies gilt auch für HO und VE.

Beim Start eines BasiCode-Programms wird der Computer in den Textbetrieb gesetzt (Initialisierung mit dem unbedingten Sprung "GOTO 20"). In dieser Betriebsart sind auf dem Bildschirm im Regelfall 24 Zeilen mit 40 Zeichen/Zeile darstellbar. Da manche Computer bis zu 80 Zeichen/Zeile abbilden können, ist diesem Umstand evtl. Durch das Programm Rechnung zu tragen.

Die Belegung der Zeile 25 und die der letzten Position einer Bildschirmzeile ist zu vermeiden. Unter Umständen kann dies zu einem Scrollen des Bildschirms und zu einem Versetzen des Cursors in die nächste Bildschirmzeile führen.

Die Position des Cursors wird durch die Variablen HO (0 ... 39) und VE (0 ... 24) und die Subroutine 110 bestimmt.

Mit der Initialisierung, d.h. vor Beginn des eigentlichen Programms, werden den Variablen HO und VE die je nach Computermodell möglichen maximalen Werte zugewiesen (z.B. HO=39 und VE=24). Die Zählung auf dem Bildschirm beginnt links oben (HO=0, VE=0).

Auf dem Amstrad PCW ist nur eine monochrome Zeichendarstellung möglich (Zeichen - hell; Hintergrund - dunkel).

GOSUB 110

Die Subroutine 110 positioniert den Cursor; sie entspricht damit den Anweisungen "LOCATE (X,Y)" oder "PRINT AT X,Y". Sinnvoll ist dieses Unterprogramm in Verbindung mit PRINT- oder INPUT-Anweisungen oder mit der Aufforderung zur Tastatur-Betätigung über die Subroutine 210.

Werden den Variablen HO und VE unzulässige Werte zugewiesen (z.B. HO>39 oder VE>24), so wird der Cursor willkürlich gesetzt, z.B. in die Mitte des Bildschirms!

```
----:----I----:----I----:----I----:----I
programs. The Netherlands National
Broadcasting service has a weekly
computer show (HOBBYSCOOP?) that
includes data broadcasts on AM radio of
BASIC programs. The idea is that you
record the broadcast programs on a
cassette recorder and then load them
into your computer. Data is broadcast at
1200 bps.
BASICODE is a "subset" of BASIC that
should allow programs written in the
BASICODE subset to be executed on nearly
any of the personal computers that have
a Microsoft flavoured BASIC on them.
This certainly includes things like
Apples, Commodores, Ataris and of course
CP/M and PC/MSDOS computers, and no doubt
many, many others. So BASICODE is also a
way of ensuring portability of BASIC
programs between widely differing
machines.#
----:----I----:----I----:----I----:----I
nach? folgt Kommando  (H=Hilfe)
```

GOSUB 120

Über diese Subroutine kann die momentane Cursorposition abgefragt werden. Deren Stellung ergibt sich aus den Variablen HO und VE. In Verbindung mit der Subroutine 110 kann die Bildschirmausgabe gesteuert werden.

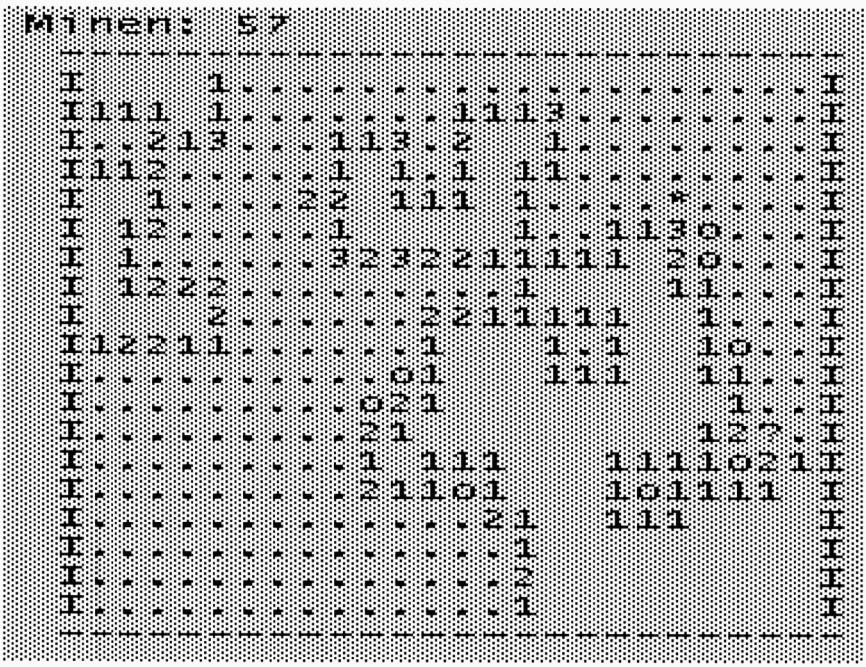
GOSUB 150

Das Unterprogramm 150 erlaubt die reverse ("auffallende") Darstellung eines Text-Strings. Der String (A$="Titel") wird mit PRINT A$ in üblicher Form dargestellt. Nach Zuweisung an SR$ werden am Beginn und Ende je drei Leerzeichen zugefügt, die Darstellung erfolgt revers. Die Stringlänge vergrößert sich um sechs Zeichen. Der Cursor steht nach dem dritten Leerzeichen nach A$ in der gleichen Zeile; um ihn in die nächste Zeile zu setzen, muß eine PRINT-Anweisung folgen. Ggf. kann der Cursor mit GOSUB 110 neu positioniert werden.

Beispiel:

```
nnnn A$="Titel"
nnnn HO=10:VE=3:GOSUB 110
nnnn SR$=A$:GOSUB 150:PRINT
```

Es bestehen aber noch weitere Möglichkeiten. Ein Beispiel:
CC(0)=4:CC(1)=1:SR$="TEST":GOSUB 150
Damit wird das Wort TEST in Blau auf einem grünen Untergrund ausgegeben. Für den weiteren Ablauf des Programms bleibt es bei den vor der letzten GOSUB 100-Anweisung mit CC(0) und CC(1) eingestellten Farben. Änderungen haben bis zum nächsten GOSUB 100 keinen Einfluß auf die normale Zeichen-Darstellung.



GOSUB 200, GOSUB 210 – Tastatur

Zur Abfrage der Tastatur stellt BasiCode-3 zwei Routinen zur Verfügung, die sich in ihrer Wirkung auf den Lauf des Programmes unterscheiden. Sie entsprechen den BASIC-Anweisungen GET und INKEY$ o.ä.

Die Subroutine 200 registriert, ob während des Programmlaufs - d.h. während der Wirksamkeit dieses Unterprogramms - eine Taste gedrückt wurde; das Programm läuft weiter. Im Gegensatz dazu hält die Subroutine 210 den Programmlauf bis Tastaturbetätigung an.

In beiden Fällen wird den Variablen IN und IN$ ein Wert zugewiesen:

IN$ - das der Taste entsprechende Zeichen (als String) und
IN - "echter" ASCII-Wert dieses Zeichens.

Wurde keine Taste betätigt (Subroutinen 200, 450), so haben die Variablen die Werte: IN$=leer; IN=0.
IN kann Werte von 32 ... 95 annehmen; Groß- und Kleinbuchstaben werden durch die gleichen ASCII-Werte repräsentiert (ASCII 65 => "A" oder "a").
Außerdem gelten:

ASCII 13 - Return / Enter,
29 ... 31 - Cursor-Steuerung,
127 - Delete.

Wurde eine Nicht-ASCII-Taste gedrückt, wird der Variablen IN eine negative Zahl zugewiesen.
In der Version -3C können auch die – sofern vorhandenen - Funktionstasten betätigt und zur Steuerung des Programmablaufs verwendet werden. Über die o.a. Routinen wird IN$ = "" und für F1 : IN = -1, F2 : IN = -2, F3 : IN = -3 usw. zurückgegeben.

Häufig wird die Subroutine 210 angewendet, um durch Betätigung einer Taste den Programmlauf zu steuern (Menu-Auswahl, Ende des Programms usw.). Als Vorteil stellt sich dar, daß die ASCII-Werte der Variablen IN keinen Unterschied zwischen Klein- und Großbuchstaben machen. Es genügt die Abfrage der Variablen IN (I); das durch Tastendruck dargestellte Zeichen muß nicht über IN$ abgefragt werden (II).

In den folgenden Beispielen soll IN bzw. IN$ auf "J/N" (ja/nein) abgefragt werden.

```
I  nn10 GOSUB 210
   nn20 IF IN=74 THEN...
   nn30 IF IN=78 THEN...
   nn40 GOTO nn10

II nn10 GOSUB 210
   nn20 IF (IN$="J")OR(IN$="j") THEN...
   nn30 IF (IN$="N")OR(IN$="n") THEN...
   nn40 GOTO nn10
```

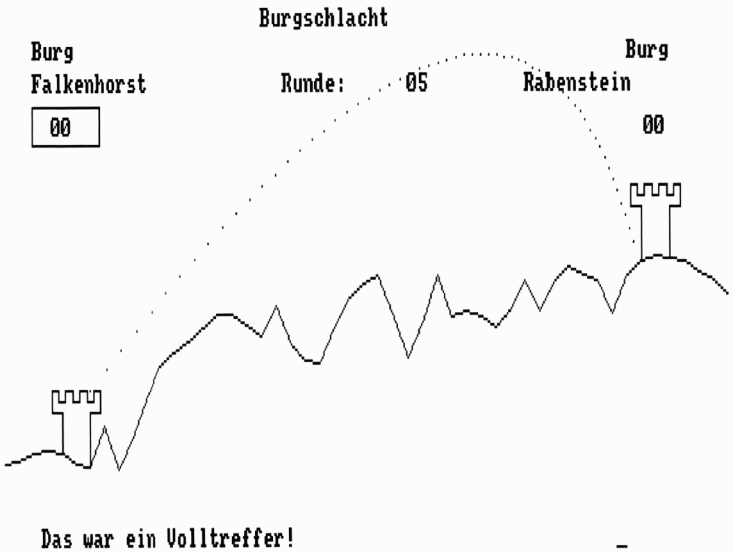
Die Verringerung des Aufwandes ist deutlich.

Eine andere Verwendung der Subroutine 210 ist die Nachbildung der INPUT-Anweisung in der Form, daß auch Anführungszeichen, Doppelpunkt und Komma direkt eingegeben und einer String-Variablen zugewiesen werden können.

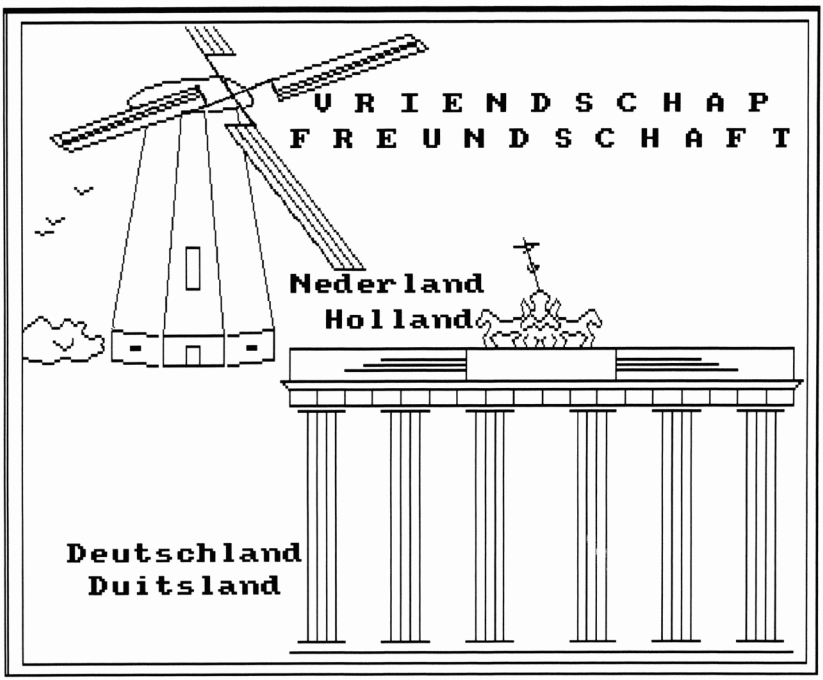
GOSUB 220
Mit der Subroutine 220 wird der ASCII-Wert eines in der Position HO, VE dargestellten Zeichens an die Variable IN zurückgegeben. IN nimmt nur Werte von 32 - 95 an. *Zwischen Klein- und Großbuchstaben kann durch Auswertung der Variablen CN (Wert computerabhängig, in den Subroutinen festgelegt) unterschieden werden. Zu berücksichtigen bleibt auch die für einzelne Computer unterschiedliche Codierung der Zeichen (z.B. Commodore).*
Eine mögliche Anwendung ist eine Hardcopy des Textbildschirms (s. S. 23).

7. Grafikmodus

GOSUB 600 löscht den Bildschirm und schaltet den Grafikmodus ein. *Der Bildschirm zeigt nach dem Löschen die Farbe, die in CC(1) codiert ist.* Die Befehle unter GOSUB 110, 120, 150, 220 sowie PRINT, INPUT und File-Arbeit sind nicht zulässig! Für Textdarstellung ist der Textmodus mit GOSUB 100 einzuschalten (Grafik ausschalten).
Für die Koordinatenaufteilung wird unter BasiCode der Punkt 0:0 oben links festgelegt. Die Mitte ist 0,5:0,5 und unten rechts ist 1:1 (zulässig ist nur <1). Die Werte werden in HG / VG übernommen.
Das Längen-Höhen-Verhältnis ist 4:3!



GOSUB 620 setzt einen Punkt entsprechend an HO/VE (DRAW / PSET / PLOT) und setzt den Grafikcursor an diese Stelle.
GOSUB 630 zeichnet eine Linie vom Punkt des aktuellen Grafikcursors zum Punkt HO/VE und setzt den Grafikcursor dorthin (LINE).
GOSUB 650 Darstellung von Text im Grafikmodus. Der Text wird vorher in SR$ übergeben. Der Grafikcursor steht an der linken oberen Ecke des Schriftbeginns. Der Text darf nicht über den rechten Rand reichen. Anschließend steht der Cursor an der in HO, VE definierten Position.
Für die Grafikbefehle kann über CN ein Farbwert übergeben werden. CN=0 ist Vordergrundfarbe und CN=1 ist Hintergrundfarbe.
Für das Programmieren in BasiCode-3C gilt: CN=0 - Grafik / Text wird in der Farbe dargestellt, die in CC(0) festgelegt ist. CN=1 - Grafik / Text wird gelöscht, d.h. in der Farbe dargestellt, die vor dem letzten GOSUB 600 in CC(1) codiert war.



8. Farben

Wie schon in der Version -3 werden die hier notwendigen Anweisungen durch Subroutinen ersetzt. Es sind folgende Farben zugelassen:

Code	Farbe	Code	Farbe
0	Schwarz	4	Grün
1	Blau	5	Hellblau (Cyan)
2	Rot	6	Gelb
3	Violett (Magenta)	7	Weiß

Farben in BasiCode-3C

Die Code-Ziffern werden den Variablen CC(0) und CC(1) zugewiesen; als Default-Einstellung gilt:

CC(0)=7 > Zeichenfarbe - Weiß
CC(1)=0 > Hintergrund - Schwarz

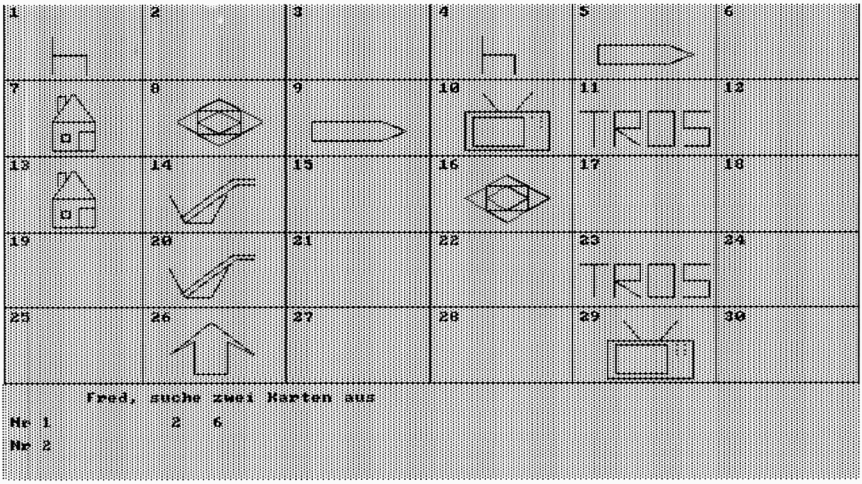
Bei Start des Programms werden diese Werte mit der Subroutine 100 übernommen und die Farben entsprechend gesetzt.

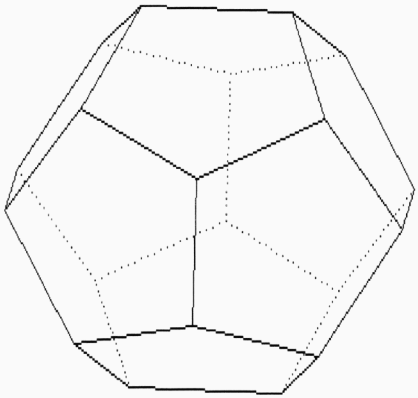
Sie gelten im Ablauf des Programms bis zu einer Änderung und der nächstfolgenden GOSUB-100-Anweisung. Die Programmzeile

CC(0)=2:CC(1)=6:GOSUB 100

liefert nach Löschen des Schirmes rote Zeichen auf gelbem Hintergrund.

Um einen hinreichenden Kontrast der Darstellung zu erreichen, empfiehlt sich ein Unterschied von "4" zwischen den Variablen in CC(0) und CC(1). Der Variablenname "CC" gilt im Sinne des BasiCode-Protokolls als verboten!





9. Sound

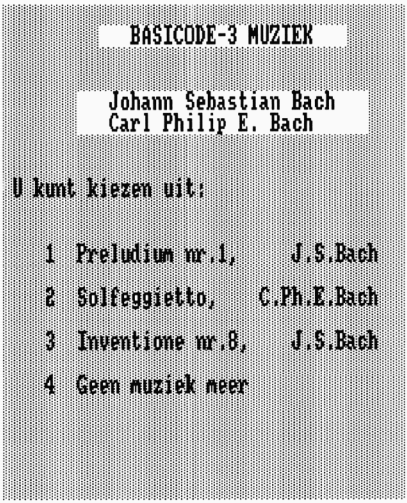
GOSUB 250 - Signalton
Diese Subroutine gibt ein akustisches Signal (entspricht ASCII 7 - BEL). Damit kann - sofern erforderlich - die Aufmerksamkeit auf einen bestimmten Schritt im Programm gelenkt werden. Allerdings ist das Signal nur von kurzer Dauer. Oft ist es notwendig, das Signal andauern zu lassen, um dann den Programmlauf nach Betätigen einer Taste verzweigen oder enden zu lassen.

GOSUB 400
Die Ausgabe ist auf einen Kanal beschränkt. Über drei Parameter wird eine Voreinstellung vorgenommen:

SP Tonhöhe (0-127) in Halbtonabstufungen mit 69 als Kammerton a (ca. 400 Hz)
SD Tondauer (1-255) in Stufen zu 0,1 Sekunden
SV Lautstärke (0-15)

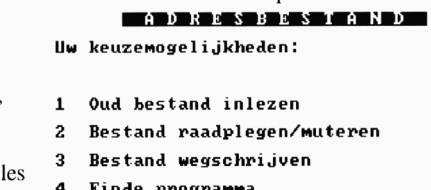
In der Regel sind mehrere Oktaven möglich, wobei der mittlere Bereich sinnvoll ist.

Da Musikstücke meist in DATA-Zeilen abgelegt werden, noch zwei Hinweise: Zeilennummern für DATAs = 25000-29999, die maximale Eingabelänge für Programmzeilen beträgt nur 60 Zeichen!



10. Arbeit mit Dateien

BasiCode-3 erlaubt es, Datenfiles auf externen Speichern (Kassette oder Diskette) anzulegen, zu schreiben und zu lesen. Das beschränkt sich jedoch auf sequentielle Files. Relative Files werden von den einzelnen Computern in zu unterschiedlicher Form verwaltet.
Damit ist es möglich, Datenfiles zwischen verschiedenen Computern - im BasiCode-Format - auszutauschen.



Die File-Verwaltung sieht vor:
- Eröffnen eines Files,
- Schreiben bzw. Lesen,
- Schließen des Files,
- Fehlerabfrage.

GOSUB 500 - Eröffnen eines Files
Mit der Eröffnung eines Files ist dessen Name in der Variablen NF$ und das angesprochene Speichermedium (Kassette, Diskette oder Microdrive) in NF festzulegen. Der Filename (NF$) kann maximal sieben Zeichen umfassen. Die Variable NF enthält den Code, der das Speichermedium festlegt und bestimmt ob das File zum Schreiben oder Lesen eröffnet wird.
Die Wirkung des der Variablen NF zuzuweisenden Codes wird im Einzelnen durch das Übersetzungsprogramm bestimmt.
Grundsätzlich gilt für die Zuweisung an NF:

Lesen	Schreiben	Speichermedium
0	1	BasiCode-Kassette
2	3	Computerspezifisch, Kassette / Diskette
4,6	5,7	Diskette

Um die Austauschbarkeit sicherzustellen, empfiehlt es sich, Datenbestände im BasiCode-Format abzulegen.
Bei Verwendung der Codeziffern 2, 4, 6 bzw. 3, 5, 7 ist zu beachten, daß es computertypische Unterschiede gibt.

n	ZX Spectrum	CBM C64
0,1	BasiCode-Kassette	
2,3 \		CBM-Kassette
4,5 >	Microdrive/Disk	Diskette
6,7 /		Diskette

Mit der folgenden Programmzeile wird ein Datenfile (zum Schreiben oder Lesen) eröffnet:

nnnn NF=n:NF$="name":GOSUB 500

Mit Abfrage der Variablen IN kann der Status, d.h. das Auftreten eines Fehlers (vgl. weiter unten) festgestellt werden.

GOSUB 580 - Schließen eines Files
Mit diesem Unterprogramm wird das mit NF=n geöffnete File geschlossen. Es genügt

nnnn GOSUB 580

ohne daß NF=n vorher explizit angegeben wird.
Sollen zur Anlage eines Files mehrere Speicher angesprochen werden (z.B. BasiCode-Kassette und Diskette), so ist das erste File zu schließen bevor das zweite eröffnet wird.

GOSUB 560 und 540 - Schreiben und Lesen eines Files
Zum Schreiben eines Datenfiles dient die Subroutine 560. Der Inhalt von SR$ wird in das File - gekennzeichnet durch NF - geschrieben. Numerische Werte sind über "GOSUB 300" oder "GOSUB 310" in die Stringvariable SR$ zu wandeln. Strings sind an SR$ zu übergeben.
Die Schreibroutine hat folgende Form:

nnnn SR=A:GOSUB 300 (oder 310)
nnnn GOSUB 560
oder nnnn SR$=A$:GOSUB 560

Das Lesen eines Files (gekennzeichnet durch NF) erfolgt über die Subroutine 540. Der gelesene Wert wird der Variablen IN$ zugewiesen, auch hier ist bei numerischen Werten und anschließenden arithmetischen Operationen eine Typwandlung über VAL(IN$) durchzuführen.
So gilt für das Lesen eines Files:

nnnn GOSUB 540:A$=IN$
oder nnnn GOSUB 540:A=VAL(IN$)

Die Variablen A$ bzw. A können dann im Programm weiter verwendet werden. Auch hier ist die Fehlerabfrage sinnvoll.

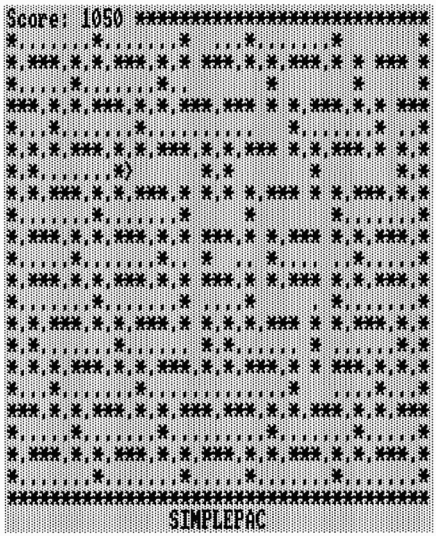
IN	Bedeutung
0	Operation fehlerfrei (o.k)
-1	Operation nicht ausgefuehrt (Fehler !)
1	EOF (End of File), nach dem Lesen des letzten Datenelements (IN$="leer")

Status- bzw. Fehlerabfrage
Bei jedem Zugriff auf externe Speicher wird der Variablen IN ein Wert zugewiesen, der zeigt, ob er fehlerfrei ablief. Vom Inhalt der Variablen IN können dann weitere Handlungen abhängig gemacht werden.

Anmerkungen:

1. In Verbindung mit dem Ansprechen externer Speicher werden u.U. die untersten Zeilen des Bildschirms für Benutzerhinweise (prompts) benötigt; diese sind deshalb freizuhalten.
2. Im grafischen Betrieb sollen die Routinen zur Fileverwaltung nicht angesprochen werden. Der Computer ist über "GOSUB 100" in den Textbetrieb zu setzen.

11. Drucken (außer Grafik)



GOSUB 350, 360 - Ausgabe über den Drucker

Grundsätzlich kann davon ausgegangen werden, daß neben der Ausgabe über den Bildschirm auch ein Ausdruck auf Papier (Erläuterungen,Tabellen usw.) sinnvoll ist. Das Programm soll also die Wahl unter beiden Möglichkeiten lassen.

Der Drucker wird über die Subroutinen 350 bzw. 360 - und nur über diese – angesprochen. Die Anweisung "GOSUB 350" entspricht der Anweisung "PRINT SR$;" - bei einer Ausgabe über den Bildschirm. Vorher ist der Inhalt der auszugebenden Variablen der Variablen SR$ zuzuweisen. Dies geschieht für

Zeichenkettenvariable (z.B. A$):

```
SR$=A$:GOSUB 350 oder
SR$="abcde":GOSUB 350
```

numerische Variable (z.B. A):

```
SR=A:GOSUB 300 (oder GOSUB 310) (Wandlung in Stringvariable SR$)
GOSUB 350
```

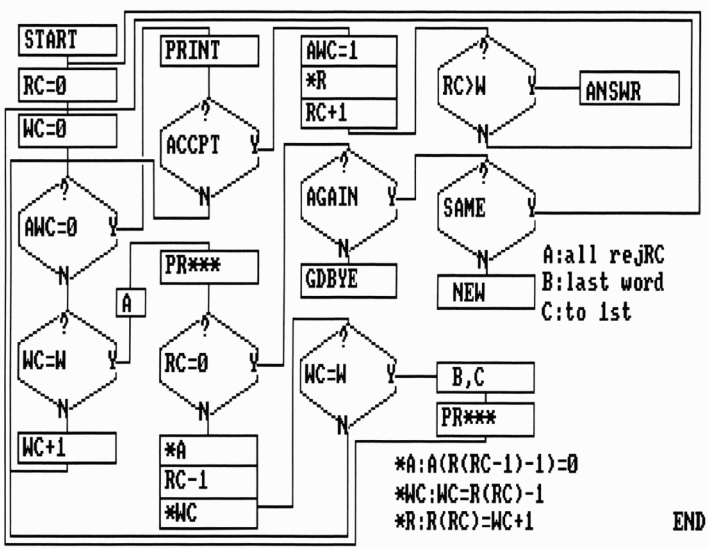
Der Ausdruck erfolgt ohne Wagenrücklauf und Zeilenvorschub; die Druckzeile wird nicht abgeschlossen. Die bewirkt die Subroutine 360; mit dieser Anweisung ist jede Anweisungsfolge zum Ausdruck einer Zeile zu beenden.

Wird die Anweisung "GOSUB 360" allein benutzt, erfolgt der Ausdruck einer Leerzeile (dies entspricht "PRINT" bei einer Ausgabe über den Bildschirm).

Als Beispiel einer Text-Hardcopy-Routine für BasiCode-3C-Programme bietet sich an:

```
1010 HT=HO:VT=VE: REM Schirmgröße
:
21000 CN=0: REM wenn bc-3c-prog. mit
21010     : REM bc-3(!)- routinen laufen
21020 FOR VE=0 TO VT:SR$=""
21030     FOR HO=0 TO HT
21040         GOSUB 220
21050         SR$=SR$+CHR$(IN+CN)
21060     NEXT HO
21070     GOSUB 350:GOSUB 360
21080 NEXT VE
21090 RETURN
```

Eine Lösung zum Erstellen einer Grafik-Hardcopy kann von BasiCode nicht bereitgestellt werden - die computerspezifische Realisierung ist ggf. in den Zeilen 20000 - 24999 abzulegen (vor der Verbreitung günstigerweise durch "REM" zu deaktivieren), wo der Nutzer eines anderen Computers stattdessen seine Routine erstellen kann.



12. Weitere Bascoder-Routinen

GOSUB 260 - Zufallszahlen
Hiermit wird der Zufallszahlengenerator des Computers aufgerufen; in RV werden dann Pseudozufallszahlen im Bereich

0 <= RV < 1

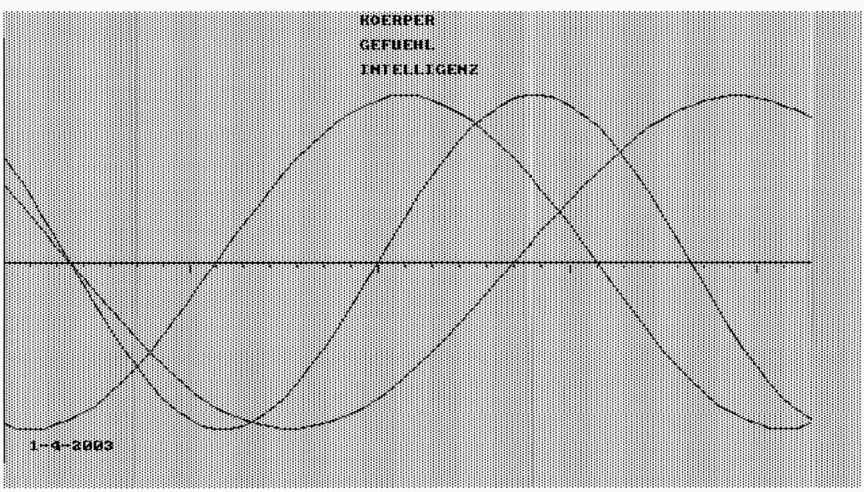
zurückgegeben. In der Regel werden nur ganzzahlige Werte benötigt, die durch Rechnung gewonnen werden können.

GOSUB 270 - Freier Arbeitsspeicher
Man kann davon ausgehen, dass BasiCode- Programme eine Länge von max. 18 KByte haben können, in Einzelfällen auch mehr. Infolge der Übertragung der einzelnen Zeichen und der rechnerinternen Umwandlung in "Token" ist die im Computer gespeicherte Programmlänge kürzer. Im Computer ist ein freier Arbeitsspeicher von mind. 16 KByte erforderlich.
Der nach Laden des Bascoders freie Speicherplatz (in Bytes) kann im Direkt-Modus mit

GOSUB 270:PRINT FR

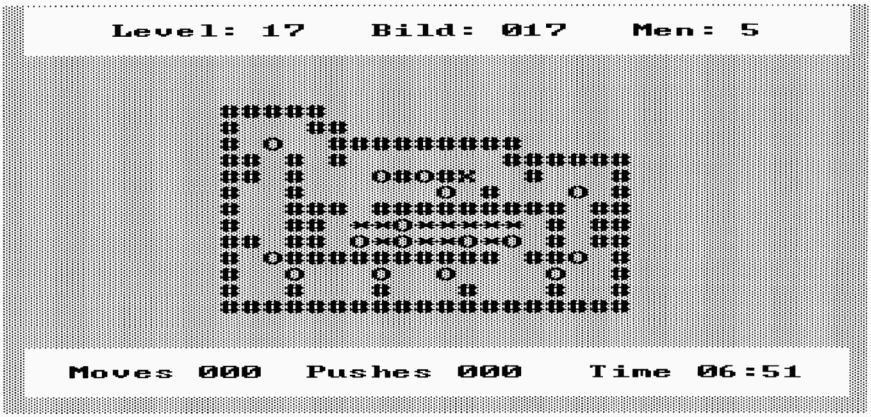
abgefragt werden. Die Wiederholung nach dem Laden des Programms und die Bildung der Differenz ergibt die computerspezifische Programmlänge (in Bytes).

GOSUB 280 - STOP-Taste
Diese Routine schaltet die Wirksamkeit einer STOP-, BREAK-, ESCAPE-Taste aus, wenn vor ihrem Aufruf der Variablen FR der Wert "1" zugewiesen wurde. Unterbrechen eines laufenden Programms mit Tastendruck ist nicht mehr möglich.
Wird FR=0 gesetzt, wird nach GOSUB 280 die STOP-Taste wieder aktiviert.



GOSUB 300 – Typwandlung
Die Subroutine 300 entspricht der in BASIC vorhandenen Funktion STR$(x), mit der numerische in Stringvariable gewandelt werden können. Damit werden die Leerräume vor und hinter numerischen Daten unterdrückt:

A$=STR$(A) wird ersetzt durch:
SR=A:GOSUB 300:A$=SR$



GOSUB 310 - Formatieren numerischer Daten
Zahlen werden in BasiCode - je nach Größe und Computer - mit 6 bzw. 9 Stellen oder in wissenschaftlicher Notation ("E-Format") angezeigt. Manche Computer (PC's) erlauben die Darstellung in "doppelter" Genauigkeit (bis zu 18 gültige Ziffern).
Die Subroutine 310 ähnelt der "PRINT USING" - Anweisung mancher BASIC-Dialekte, die jedoch vielfältiger eingesetzt werden kann als die Subroutine 310. In BasiCode-3 ist nur das Formatieren der Ausgabe numerischer Werte - über Bildschirm oder Drucker - möglich.
Die formatierte Ausgabe des Wertes der numerischen Variablen A erfolgt über die Variablen SR$, CT und CN. Dabei bedeuten:

SR numerische Variable, deren Wert in SR$ formatiert dargestellt werden soll,
CT Anzahl der Zeichen, die in SR$ enthalten sind (einschl. Dezimalpunkt und Vorzeichen),
CN Anzahl der Nachkommastellen.

Diese Variablen sind vor dem Aufruf der Subroutine 310 zu belegen.
Die Zeichenkette SR$ kann maximal nur neun Ziffern enthalten, d.h. daß CT begrenzt ist:

```
    1  - Vorzeichen,
+ vk   - Anzahl der Vorkommastellen,
+  1   - Dezimalpunkt,
+ CN   - Anzahl der Nachkommastellen,
  ---------------------------------
   CT - Anzahl der Zeichen
```

In Abhängigkeit von der Größe der darzustellenden Zahl gilt:

vk + CN <= 9 (ohne führende Null falls SR < 1)

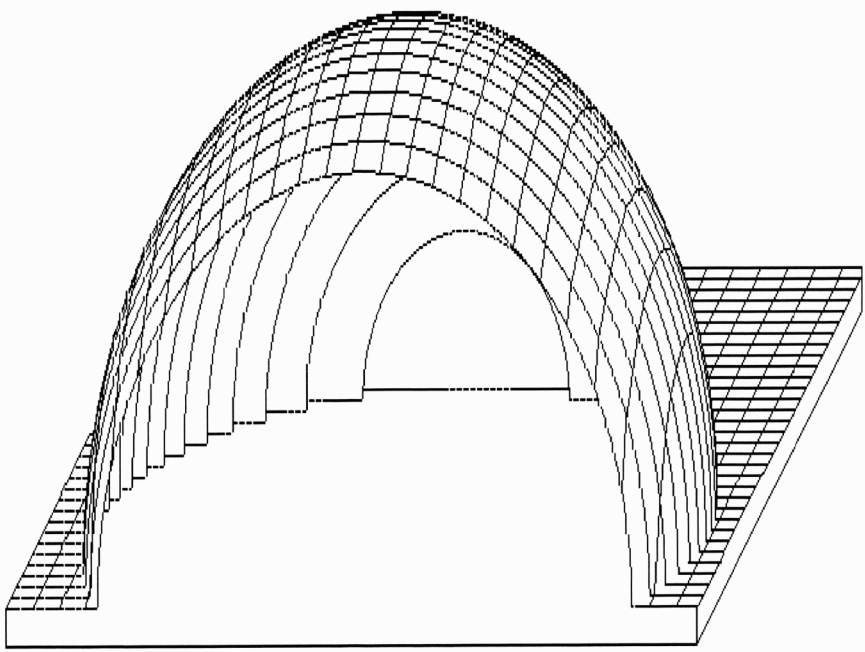
Mit der Subroutine 310 ist es nicht möglich, Zahlen im wissenschaftlichen Format darzustellen.

Kann die Zahl nicht im vorausbestimmten Format angezeigt werden, enthält SR$ Sterne ("*"). Ggf. wird die Zahl auf CN Stellen gerundet. Die Werte der Variablen CT, CN und SR werden mit dem Aufruf der Subroutine nicht verändert.
Beim Programmieren sind die Werte für CT und CN sorgfältig zu bestimmen, z.B.:
- Ganze Zahlen (-1E8 ... +1E8): CT=11
- SR < 1 (Vorzeichen, führende Null, Dezimalpunkt, neun Nachkommastellen): CT=12, N=9

Der Variablen CT kann ein Wert bis zu 20 zugewiesen werden; dies führt zu einer Positionierung der Ausgabe in der Zeile. Besser ist es, die Subroutine 110 zu benutzen.

GOSUB 330 - Klein- -> Großbuchstaben
Dieses Unterprogramm ändert alle in der Zeichenkette SR$ vorhandenen Kleinbuchstaben in Großbuchstaben, indem der ASCII-Wert der Zeichen im Bereich 96 ... 128 um 32 vermindert wird. Aus ASCII 97 ("a") wird ASCII 65 ("A"). Da das Alphabet die ASCII-Werte 97 ... 122 umfaßt, werden auch die Sonderzeichen im Bereich 123 ... 126 gewandelt.
Die Zeichen der ursprünglichen Zeichenkette werden nicht geändert.

GOSUB 450 - Warteroutine
Diese Subroutine unterbricht den Programmlauf für eine vorgegebene Zeitspanne. Durch Drücken einer Taste kann sie abgebrochen werden. Sie entspricht der PAUSE- bzw. SLEEP-Anweisung mancher BASIC-Dialekte.



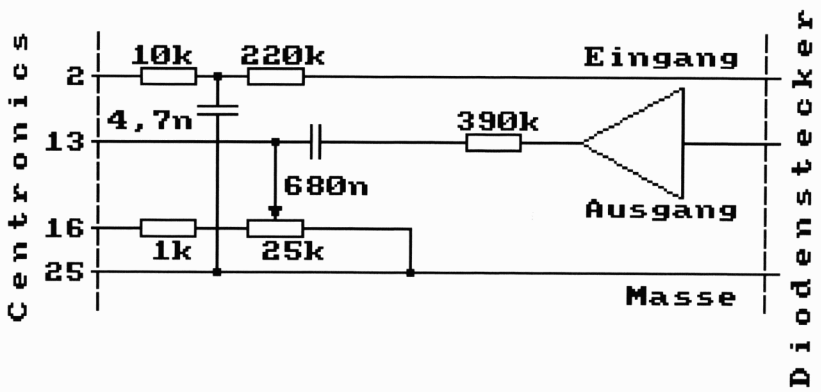
Die Wartezeit ist vor dem Aufruf der Variablen SD zuzuweisen:

$$SD = \frac{\text{Wartezeit (Sekunden)}}{0.1}$$

Wird eine Taste gedrückt, so wird die Routine abgebrochen, den Variablen IN und IN$ werden der ASCII-Wert und das Zeichen übergeben, die Variable SD enthält die Restzeit (in Einheiten von 0.1 Sekunden).
Um eine nicht abbrechbare Pause zu erreichen, wird vorgeschlagen, die Routine 400 mit dem Parameter SV=0 zu verwenden, also einen Ton der Lautstärke Null zu spielen.

13. Das BasiCode-Protokoll

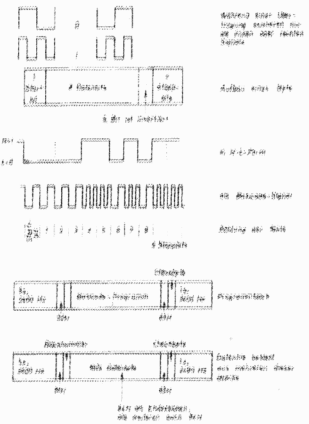
Das ursprüngliche Aufzeichnungsmedium von BasiCode war die Kassette. Bei Computern, die auf jeden Fall über Diskettenlaufwerke verfügen, wurde die Arbeit mit Kassetten in separate Programme außerhalb des Bascoders verlegt (CP/M: BCREAD.COM, BCWRITE.COM - DOS: BASICODE.COM oder BASICODE.EXE), ein Interface zum Anschluß des Kassettengerätes an den Parallelport müßte zur Verfügung stehen.



Die Aufzeichnung erfolgt mit einer 1200-Hz-Schwingung für ein 0-Bit und zwei 2400-Hz-Schwingungen für ein 1-Bit.
Programme werden, da unterschiedliche Computer unterschiedliche Token verwenden, als ASCII-Listings und in einem Zug übertragen. Außerdem gibt es die Möglichkeit, Daten (Binärfiles) zu übertragen, hier können alle Werte von 0 bis 255 auftreten, daher erfolgt dann eine Gliederung in Blöcke zu 1024 Byte.
Die Übertragung jedes Bytes (mit dem niederwertigsten Bit zuerst) erfolgt asynchron mit einem Start- und zwei Stoppbit. Zusäzlich wird jedes achte Datenbit invertiert, das läuft (bei Programmübertragungen) auf ein drittes Stoppbit hinaus und erhöht damit zusätzlich die Datensicherheit, da das 8. Bit bei ASCII-Zeichen immer 0 ist.

Eine Programmübertragung hat folgenden Aufbau:

- ca. fünf Sekunden Startton (2400 Hz) zum Synchronisieren
- ein Startbyte (STX, start of text), durch Invertierung des 8. Bits 82H
- das eigentliche Programm (ASCII-File, einzelne Zeilen durch ENTER getrennt, wegen der Invertierung des 8. Bits 8DH)
- ein Stopbyte (ETX, end of text), mit invertiertem 8. Bit 83H
- ein Checkbyte (Prüfsumme)
- ca. eine Sekunde Abschlußton (2400Hz)



Ein Datenfile ist wie folgt aufgebaut:

- fünf Sekunden 2400 Hz
- ein Startbyte (SOH, start of header), 81H nach Invertierung des 8. Bits
- Blocknummer von 0 (80H mit invertiertem 8. Bit) beginnend
- 1024 Datenbyte (mit jeweils invertiertem 8. Bit)
- Ende-Byte (ETX, end of text), 83H nach Invertieren des 8. Bits
- ein Checkbyte (Prüfsumme)
- eine Sekunde Abschlußton (2400 Hz)
- beim letzten Datenblock, weil er kürzer als 1024 Byte sein kann, ein Byte (EOT, end of transmission), nach Invertierung des 8. Bits 84H und weitere 84H , bis die Länge 1024 erreicht ist

Selbstverständlich lassen sich BasiCode-Programme auch per Diskette (sowie durch serielle Kopplung, DFÜ, Internet oder SD-Cards, USB-Sticks usw.) übertragen, findige Bastler haben inzwischen Lösungen für viele historische Computer gefunden, außerdem gibt es zu Emulatoren meist auch passende Begleitprogramme – in jedem Fall ist hier jedoch peinlich darauf zu achten, die ASCII-Form des Abspeicherns zu verwenden - das computerinterne verkürzte Speicherformat (Token) ist nur für Computer des selben Typs lesbar.

14. weblinks

https://de.wikipedia.org/wiki/BASICODE
ausführlicher Wikipedia-Artikel

http://basicode.de/
Bascoder für verschiedene Computer sowie viele Programme

https://github.com/robhagemans/basicode
viele Programme aus verschiedenen Quellen

http://robhagemans.github.io/basicode/
BasiCode unmittelbar im webbrowser ausführen

http://www.kc85emu.de/scans/rfe0190/Basicode.htm
Aufzeichnungsformat (Auszug aus dem DDR-BasiCode-Buch)

https://groups.yahoo.com/neo/search?query=Basicode
eine Yahoo-Gruppe

15. Ein Mini-Wörterbuch

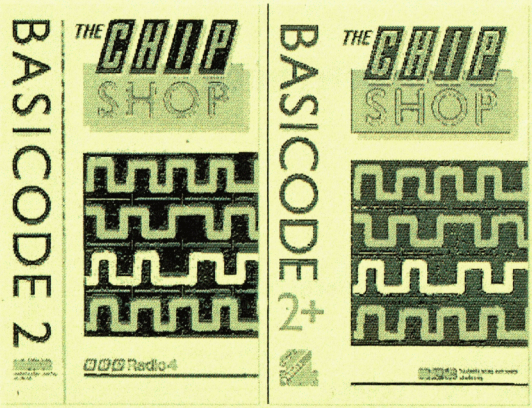
BESTAND	Datei
DUUR	Dauer
FOUT	Fehler
FOUTE INVOER	falsche Eingabe
GEEF UW KEUZE	Bitte wählen Sie
GELUID	Sound
GRAFIEK OP HET SCHERM	Grafik auf dem Bildschirm
GRAFISCH BEDRIJF	Grafik-Betriebsart
HOOFDLETTER	Großbuchstabe
HOOFDPROGRAMMA	Hauptprogramm
KLAAR	Löschen (vgl. CLEAR)
KLEUR	Farbe (vgl. COLOR)
PAGINA	Seite (vgl. PAGE)
REGEL	Programmzeile
SPATIEBALK	Leertaste
SWART	schwarz
TEKEN LIJNSTUK	ein Geradenstück zeichnen
TOETS	Taste
TOT ZIENS	bis bald
UITLEG	Erklärung
U KUNT KIEZEN UIT :	Sie können wählen aus :
WISSEN	löschen
WIT	weiß

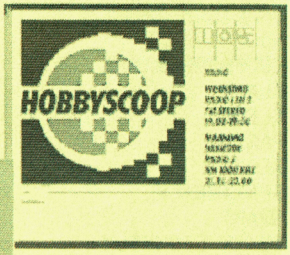
16. Literatur

Michael Wiegand; Manfred und Heike Fillinger:
Basicode. Otto Maier Verlag Ravensburg 1984 ISBN 3-473-44010-8
auf beiliegender Kassette Bascoder für Apple II + IIe, BBC Modell A + B,
Colour Genie, Commodore 3000, 4000 + 8000, C 64, PET 2001, VC 20, DAI, Dragon 32,
Junior (elektor), erw. Version mit 9KB-Basic, Junior mit VDU-Karte, Sharp MZ 80 A,
Sharp MZ 80 K, ZX 81, ZX Spectrum, TRS 80 Modell I od. III/Videogenie
und fünfzehn BasiCode-Programme

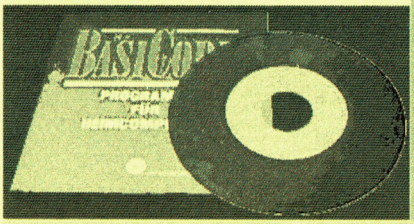
Hermine Bakker, Jacques Haubrich (Autoren), Stichting BASICODE (Herausgeber):
Het BASICODE-3 boek, Buch und Kassette, Kluwer Technische Boeken BV Verlag,
Deventer-Antwerpen, Belgien, ISBN 90-201-2111-1, NUGI 434/857

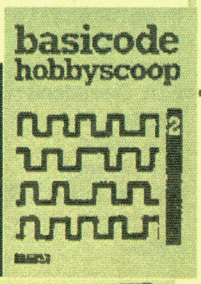
Horst Völz /federführender Autor):
BasiCode. Verlag Technik, Berlin 1990 Bestellnummer: 554 342 0 ISBN 3-341-00895-0
auf beiliegender Schallplatte Bascoder für KC 85/3 & 85/2, KC 85/4, Commodore C-64,
CPC-464, -664, -6128, KC compact, KC 87, KC 85/1, Z9001, Z1013,
C plus 4 & C 16 (64 K) und AC 1
sowie mehrere Programmlistings im Buch

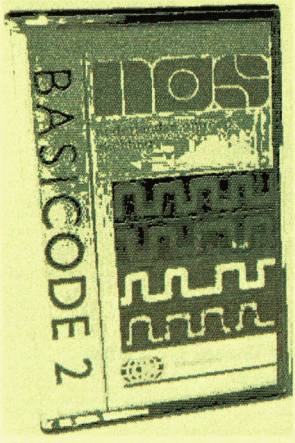


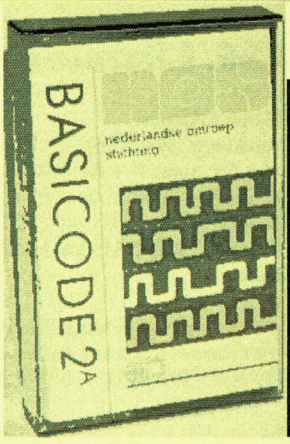


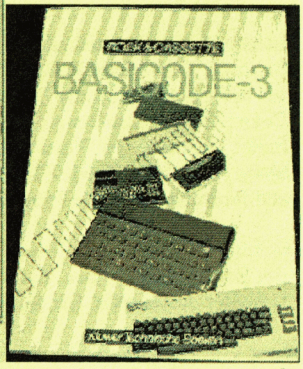
BasiCode auf unterschiedlichen Medien und aus unterschiedlichen Ländern











Thunderbirds & Unlimited
Paperware